Wireless Sensor Networks

Prakhar Banga
prakban@iitk.ac.in

Vineet Hingorani
viner@iitk.ac.in

Prof. Sumit Ganguly
sganguly@iitk.ac.in

Dept. of CSE, I.I.T. Kanpur

April 15, 2013

Sensors

Sensors

- A device with sensing capability:

Sensors

- A device with sensing capability:
  - Thermometer

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone(Sound)

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone(Sound)
- Wireless sensor nodes contain:

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone(Sound)
- Wireless sensor nodes contain:
  - A sensor/sensors

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone(Sound)
- Wireless sensor nodes contain:
  - A sensor/sensors
  - Microprocessor

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone(Sound)
- Wireless sensor nodes contain:
  - A sensor/sensors
  - Microprocessor
  - Storage

Sensors

- A device with sensing capability:
  - Thermometer(Temperature)
  - Hygrometer(Humidity)
  - Microphone(Sound)
- Wireless sensor nodes contain:
  - A sensor/sensors
  - Microprocessor
  - Storage
  - Wirless module

Wireless Sensor Networks

Wireless Sensor Networks

- Networks built for sensor applications

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring
- Examples:

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring
- Examples:
  - Forest Fire detection

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring
- Examples:
  - Forest Fire detection
  - Activity monitoring(Security)

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring
- Examples:
  - Forest Fire detection
  - Activity monitoring(Security)
  - GPS tracking

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring
- Examples:
  - Forest Fire detection
  - Activity monitoring(Security)
  - GPS tracking
  - Plant health monitoring

Wireless Sensor Networks

- Networks built for sensor applications
- Consist of sensor nodes and gateway nodes
- Nodes may/may not be capable of computations
- Usually used for monitoring
- Examples:
  - Forest Fire detection
  - Activity monitoring(Security)
  - GPS tracking
  - Plant health monitoring
  - ...and various others

Wireless Sensor Networks

Differences with ad-hoc networks

Wireless Sensor Networks

Differences with ad-hoc networks

- Dense deployment

Wireless Sensor Networks

Differences with ad-hoc networks

- Dense deployment
- Large number of nodes

Wireless Sensor Networks

Differences with ad-hoc networks

- Dense deployment
- Large number of nodes
- Unreliable nodes

Wireless Sensor Networks

Differences with ad-hoc networks

- Dense deployment
- Large number of nodes
- Unreliable nodes
- Broadcast/Multicast paradigm

Wireless Sensor Networks

Differences with ad-hoc networks

- Dense deployment
- Large number of nodes
- Unreliable nodes
- Broadcast/Multicast paradigm
- Limited node capability

Wireless Sensor Networks

Differences with ad-hoc networks

- Dense deployment
- Large number of nodes
- Unreliable nodes
- Broadcast/Multicast paradigm
- Limited node capability
- Addressing issues

Wireless Sensor Networks

Deployment And Maintanence

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

- System development from scratch

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

- System development from scratch
- Nodes are programmed individually

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

- System development from scratch
- Nodes are programmed individually
- Installed at various locations

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

- System development from scratch
- Nodes are programmed individually
- Installed at various locations
- Small changes done manually:

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

- System development from scratch
- Nodes are programmed individually
- Installed at various locations
- Small changes done manually:
  - Changes in logical structure of network

Wireless Sensor Networks

Deployment And Maintanence

Current state-of-art:

- System development from scratch
- Nodes are programmed individually
- Installed at various locations
- Small changes done manually:
  - Changes in logical structure of network
  - Reprogramming of nodes

Problem Statement

Problem Statement

Designing a general framework:

Problem Statement

Designing a general framework:

- General system architecture to suit all needs

Problem Statement

Designing a general framework:

- General system architecture to suit all needs
- Highly flexible, supports remote changes

Problem Statement

Designing a general framework:

- General system architecture to suit all needs
- Highly flexible, supports remote changes
- Should be lightweight, secure, reliable, fault-tolerant

Problem Statement

Designing a general framework:

- General system architecture to suit all needs
- Highly flexible, supports remote changes
- Should be lightweight, secure, reliable, fault-tolerant
- Analogous to systemization of database(1970s)

Problem Statement

Constraints

Problem Statement

Constraints

Hardware constraints:

Problem Statement

Constraints

Hardware constraints:

- Costs involved

Problem Statement

Constraints

Hardware constraints:

- Costs involved
- Energy consumption

Problem Statement

Constraints

Hardware constraints:

- Costs involved
- Energy consumption
- Communication range

Problem Statement

Constraints

Hardware constraints:

- Costs involved
- Energy consumption
- Communication range
- Data transfer rate

Problem Statement

Constraints

Hardware constraints:

- Costs involved
- Energy consumption
- Communication range
- Data transfer rate
- Limited storage

Problem Statement

Constraints

Hardware constraints:

- Costs involved
- Energy consumption
- Communication range
- Data transfer rate
- Limited storage
- Processing power

Problem Statement

Constraints

Problem Statement

Constraints

Design constraints:

Problem Statement

Constraints

Design constraints:

- Application dependent:

Problem Statement

Constraints

Design constraints:

- Application dependent:
  - Mobile vs. fixed nodes

Problem Statement

Constraints

Design constraints:

- Application dependent:
  - Mobile vs. fixed nodes
  - Level of security

Problem Statement

Constraints

Design constraints:

- Application dependent:
  - Mobile vs. fixed nodes
  - Level of security
  - Central vs. Distributed processing

Problem Statement

Constraints

Design constraints:

- Application dependent:
  - Mobile vs. fixed nodes
  - Level of security
  - Central vs. Distributed processing
- Topology dependent:

Problem Statement

Constraints

Design constraints:

- Application dependent:
  - Mobile vs. fixed nodes
  - Level of security
  - Central vs. Distributed processing
- Topology dependent:
  - No. of sensor vs. gateway nodes

Problem Statement

Constraints

Design constraints:

- Application dependent:
  - Mobile vs. fixed nodes
  - Level of security
  - Central vs. Distributed processing
- Topology dependent:
  - No. of sensor vs. gateway nodes
  - Personal area vs. Wide area

Related Work

Related Work

- Use of TinyOS is prevalent

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets
- Most of these use Berkeley Motes

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets
- Most of these use Berkeley Motes

Disadvantages:

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets
- Most of these use Berkeley Motes

Disadvantages:

- Hardware Costs suffer a lot

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets
- Most of these use Berkeley Motes

Disadvantages:

- Hardware Costs suffer a lot
- TinyOS is really not ‘tiny’

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets
- Most of these use Berkeley Motes

Disadvantages:

- Hardware Costs suffer a lot
- TinyOS is really not ‘tiny’
- No capability to map virtual code to binary

Related Work

- Use of TinyOS is prevalent
- Virtual machine code as programs
- CSMA and TDMA as MAC Protocols
- Flooding the network with packets
- Most of these use Berkeley Motes

Disadvantages:

- Hardware Costs suffer a lot
- TinyOS is really not ‘tiny’
- No capability to map virtual code to binary
- Speed of networking suffers

Various Issues

Various Issues

- Programming Paradigm

Various Issues

- Programming Paradigm
  - Programming Interface

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking
  - Scheduling

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking
  - Scheduling
  - Addressing

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking
  - Scheduling
  - Addressing
- OS

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking
  - Scheduling
  - Addressing
- OS
  - Very lightweight

Various Issues

- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking
  - Scheduling
  - Addressing
- OS
  - Very lightweight
  - Modular for required changes

Various Issues

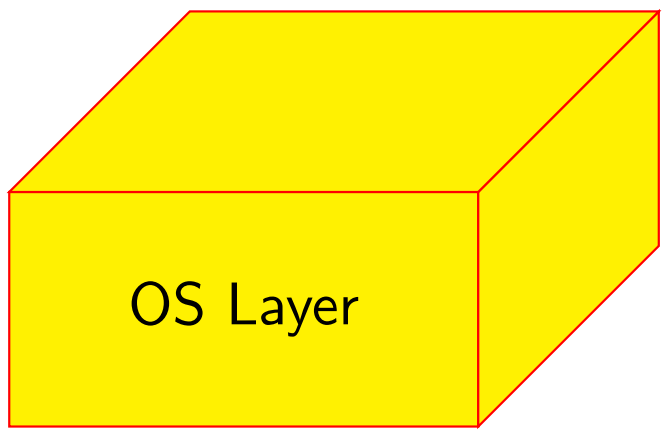
- Programming Paradigm
  - Programming Interface
  - Version Control
  - Dissemination/Acquisition
- Networking
  - Scheduling
  - Addressing
- OS
  - Very lightweight
  - Modular for required changes
  - Layered model

Proposed Architecture

Stack Model

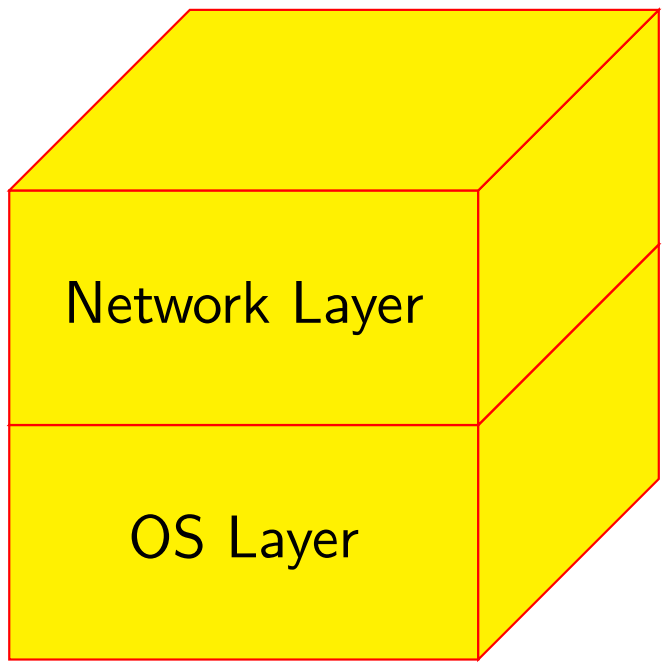
Proposed Architecture

Stack Model



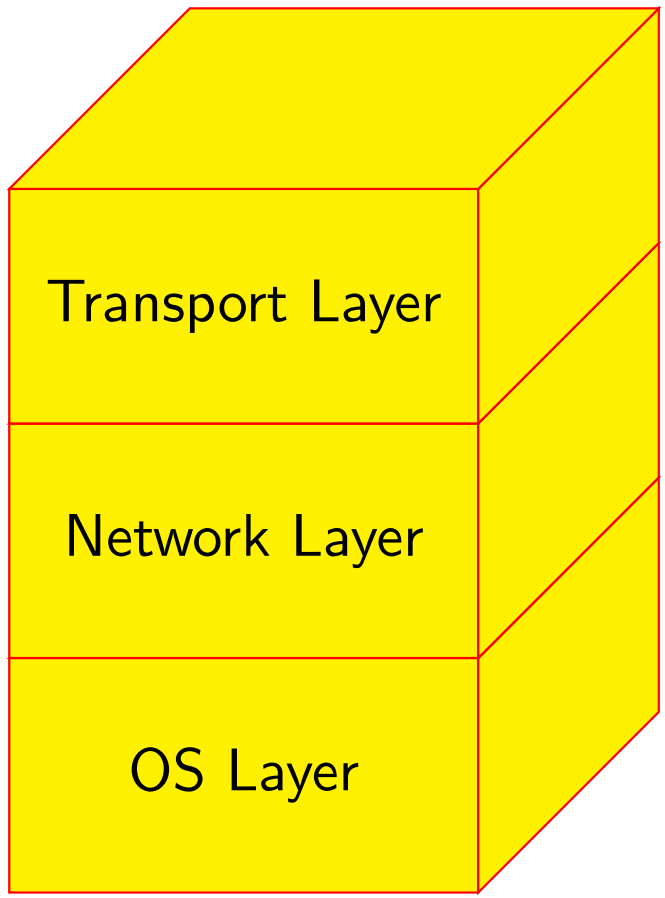
Proposed Architecture

Stack Model



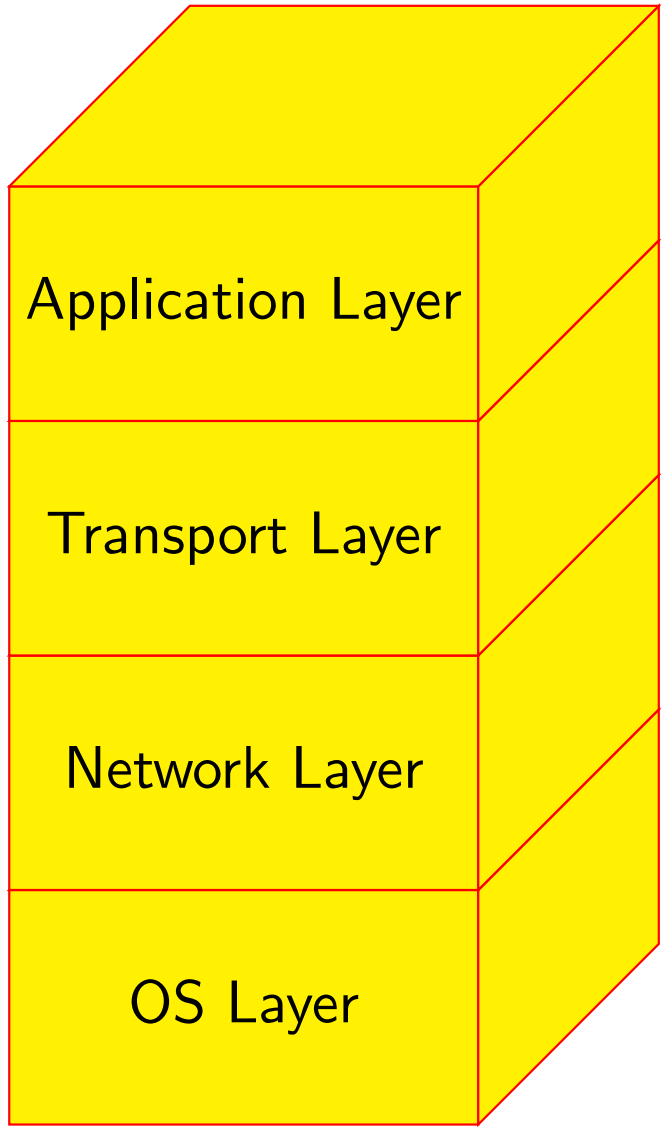
Proposed Architecture

Stack Model



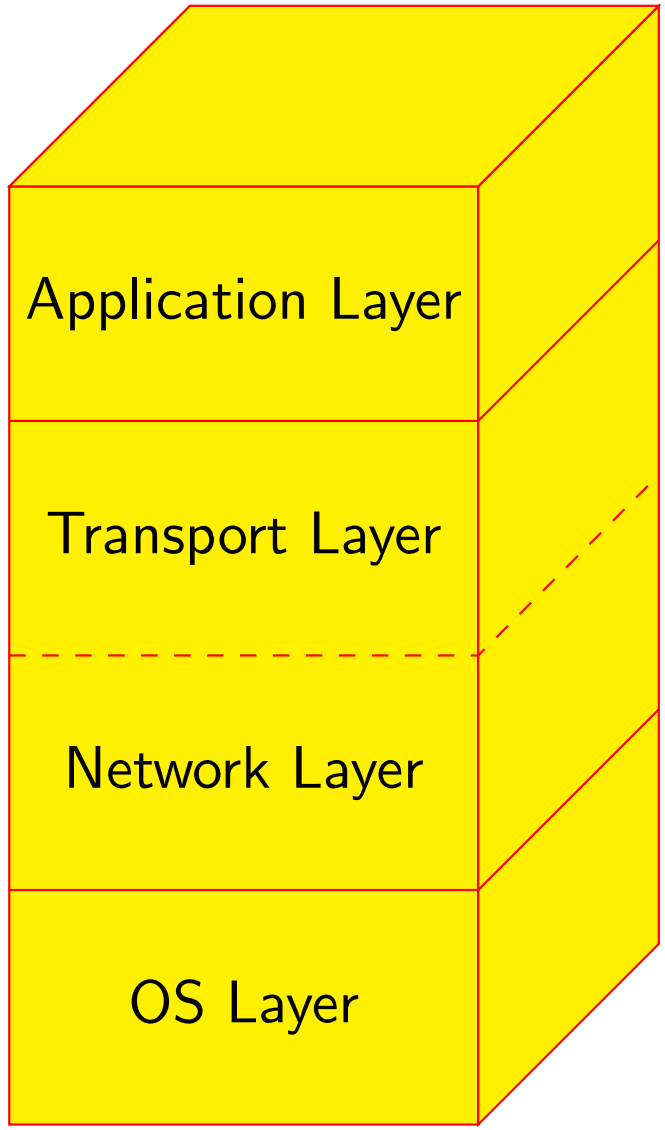
Proposed Architecture

Stack Model



Proposed Architecture

Stack Model



Proposed Architecture

Hierachical Control System

Proposed Architecture

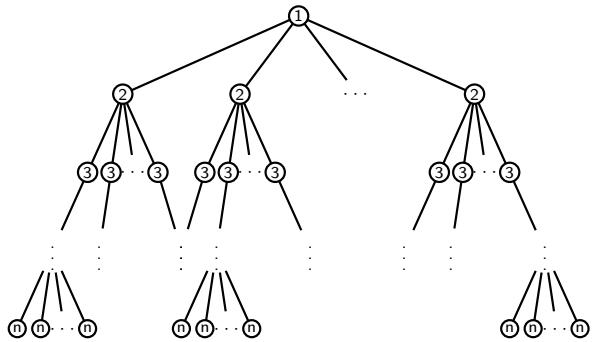
Hierachical Control System

- n layers of nodes

Proposed Architecture

Hierachical Control System

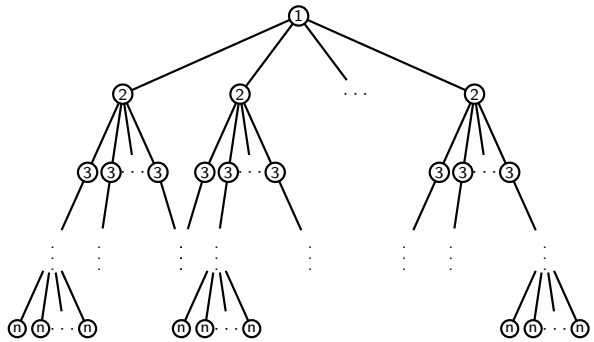
- n layers of nodes



Proposed Architecture

Hierachical Control System

- n layers of nodes

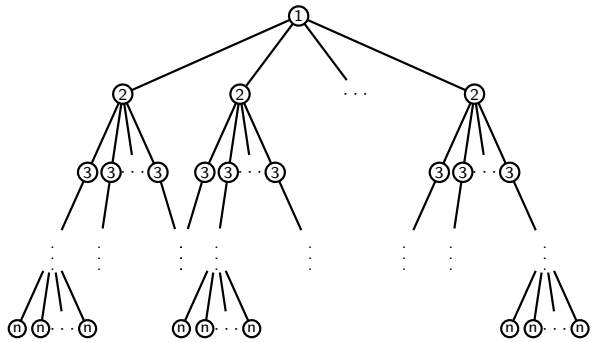


- Higher layers have greater capabilities than lower nodes

Proposed Architecture

Hierachical Control System

- n layers of nodes

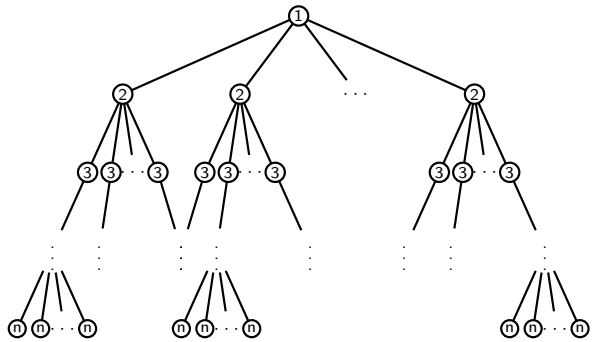


- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):

Proposed Architecture

Hierachical Control System

- n layers of nodes

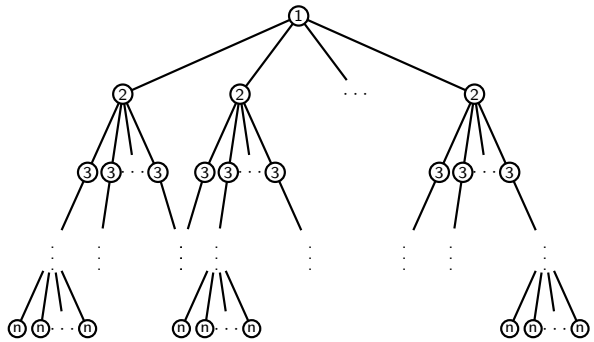


- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):
  - Assigns ‘tasks’ to its children

Proposed Architecture

Hierachical Control System

- n layers of nodes

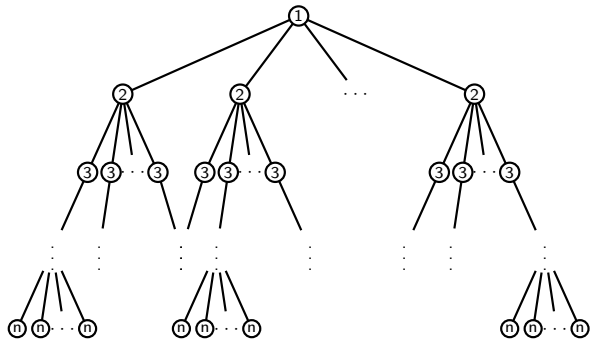


- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):
  - Assigns ‘tasks’ to its children
  - Gathers data from its children

Proposed Architecture

Hierachical Control System

- n layers of nodes

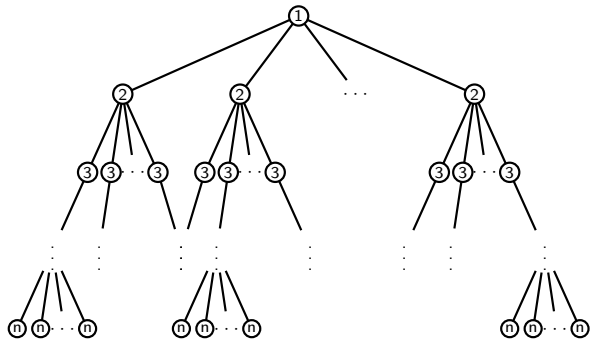


- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):
  - Assigns ‘tasks’ to its children
  - Gathers data from its children
  - Processes the gathered data and sends data back to its parent

Proposed Architecture

Hierachical Control System

- n layers of nodes

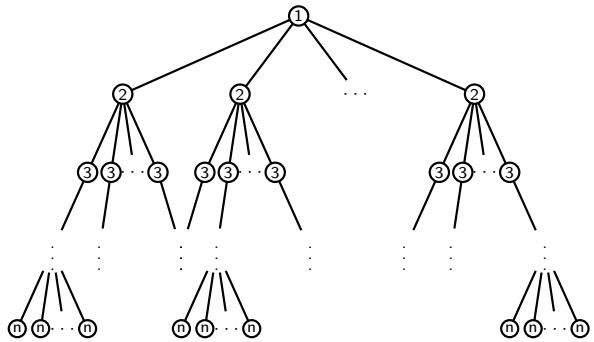


- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):
  - Assigns ‘tasks’ to its children
  - Gathers data from its children
  - Processes the gathered data and sends data back to its parent
- The tasks are:

Proposed Architecture

Hierachical Control System

- n layers of nodes

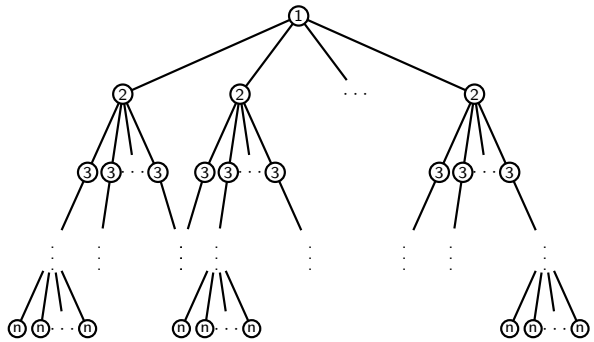


- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):
  - Assigns ‘tasks’ to its children
  - Gathers data from its children
  - Processes the gathered data and sends data back to its parent
- The tasks are:
  - Programs in an query-based language

Proposed Architecture

Hierachical Control System

- n layers of nodes



- Higher layers have greater capabilities than lower nodes
- Each node(except the lowest layer):
  - Assigns ‘tasks’ to its children
  - Gathers data from its children
  - Processes the gathered data and sends data back to its parent
- The tasks are:
  - Programs in an query-based language
  - Aimed at data filtering/aggregation

Proposed Architecture

Addressing

Proposed Architecture

Addressing

- Assumption: need not address individual nodes

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:
  - Novel Concept of Attribute-Value pairs:

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:
  - Novel Concept of Attribute-Value pairs:

Attribute	Value
A1	V1
A2	V2
A3	V3
.	.
.	.
.	.
Ak	Vk

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:
  - Novel Concept of Attribute-Value pairs:

Attribute	Value
A1	V1
A2	V2
A3	V3
.	.
.	.
.	.
Ak	Vk

  - Properties used to distribute and diffuse the tasks

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:
  - Novel Concept of Attribute-Value pairs:

Attribute	Value
A1	V1
A2	V2
A3	V3
.	.
.	.
.	.
Ak	Vk

  - Properties used to distribute and diffuse the tasks
- Examples:

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:
  - Novel Concept of Attribute-Value pairs:

Attribute	Value
A1	V1
A2	V2
A3	V3
.	.
.	.
.	.
Ak	Vk

  - Properties used to distribute and diffuse the tasks
- Examples:

Attribute	Value
Floor	3
Lab	1
Row	5

Smart Building

Proposed Architecture

Addressing

- Assumption: need not address individual nodes
- Application level adressing:
  - Novel Concept of Attribute-Value pairs:

Attribute	Value
A1	V1
A2	V2
A3	V3
.	.
.	.
.	.
Ak	Vk

  - Properties used to distribute and diffuse the tasks
- Examples:

Attribute	Value
Floor	3
Lab	1
Row	5

Smart Building

Attribute	Value
Animal	Lion
Location	Pond Area
Age of Animal	4

Animal Monitoring

Proposed Architecture

Mobility

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:
  - Based on geographic location properties

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:
  - Based on geographic location properties

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Rajasthan

Departed Jaipur

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:
  - Based on geographic location properties

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Rajasthan

Departed Jaipur

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Gujarat

Arriving Himmatnagar

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:
  - Based on geographic location properties

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Rajasthan

Departed Jaipur

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Gujarat

Arriving Himmatnagar

  - Based on node particulars

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:
  - Based on geographic location properties

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Rajasthan

Departed Jaipur

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Gujarat

Arriving Himmatnagar

  - Based on node particulars

Attribute	Value
Troop	18-Grenadier
Area	Bunker-4
Condition	Healthy

Fighting Soldier

Proposed Architecture

Mobility

- Addressing is a challenge in general mobile WSNs
- Using the addressing scheme proposed:
  - A node can change its value for the attributes
  - Signal Strength, GPS, other landmarks can be used to change the Table
- Examples:
  - Based on geographic location properties

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Rajasthan

Departed Jaipur

Attribute	Value
Vehicle	Bus
Sub-agency	Krishna
State	Gujarat

Arriving Himmatnagar

  - Based on node particulars

Attribute	Value
Troop	18-Grenadier
Area	Bunker-4
Condition	Healthy

Fighting Soldier

Attribute	Value
Troop	Gurkha
Area	Fence-8
Condition	Critical

Soldier Shot

Proposed Architecture

Query-Based System

Proposed Architecture

Query-Based System

- Each node runs a query on some data

Proposed Architecture

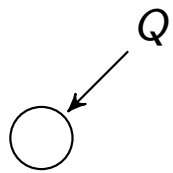
Query-Based System

- Each node runs a query on some data

Proposed Architecture

Query-Based System

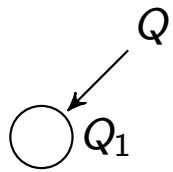
- Each node runs a query on some data



Proposed Architecture

Query-Based System

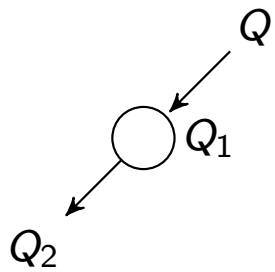
- Each node runs a query on some data



Proposed Architecture

Query-Based System

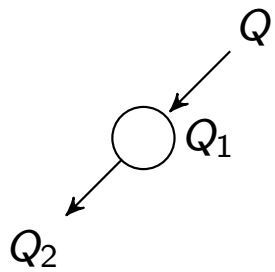
- Each node runs a query on some data



Proposed Architecture

Query-Based System

- Each node runs a query on some data

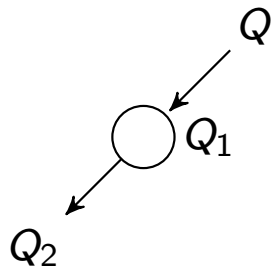


$$Q = Q_1 \circ Q_2$$

Proposed Architecture

Query-Based System

- Each node runs a query on some data



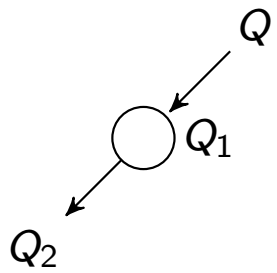
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.

Proposed Architecture

Query-Based System

- Each node runs a query on some data



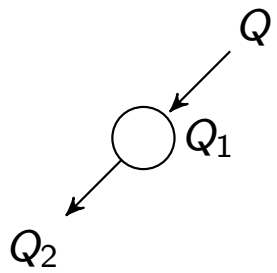
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,

Proposed Architecture

Query-Based System

- Each node runs a query on some data



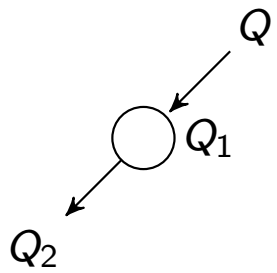
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query

Proposed Architecture

Query-Based System

- Each node runs a query on some data



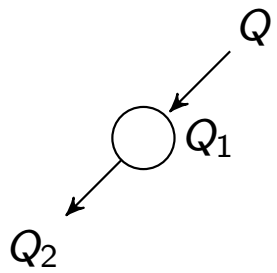
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp$>$102;

Proposed Architecture

Query-Based System

- Each node runs a query on some data



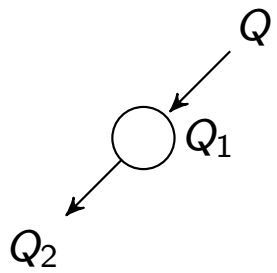
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;

Proposed Architecture

Query-Based System

- Each node runs a query on some data



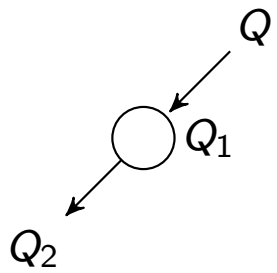
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;

Proposed Architecture

Query-Based System

- Each node runs a query on some data



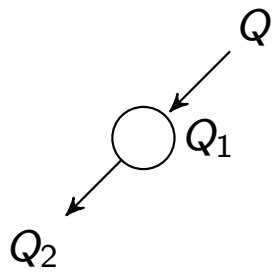
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends

Proposed Architecture

Query-Based System

- Each node runs a query on some data



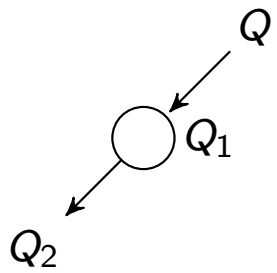
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp$>$102;
    Q_2=animal whose body_temp$>$102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)

Proposed Architecture

Query-Based System

- Each node runs a query on some data



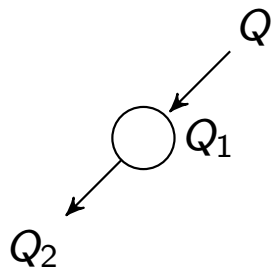
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query

Proposed Architecture

Query-Based System

- Each node runs a query on some data



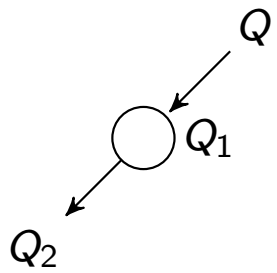
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query
    Q=animal id who ate 1 day before;

Proposed Architecture

Query-Based System

- Each node runs a query on some data



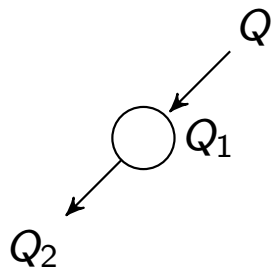
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query
    Q=animal id who ate 1 day before;
    Q_2=send eating animal's id and corresponding time;

Proposed Architecture

Query-Based System

- Each node runs a query on some data



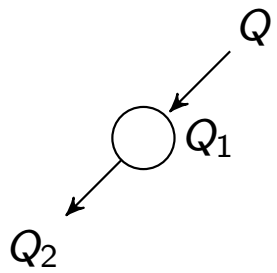
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query
    Q=animal id who ate 1 day before;
    Q_2=send eating animal's id and corresponding time;
    Q_1=store animal's last eating time and filter for sending;

Proposed Architecture

Query-Based System

- Each node runs a query on some data



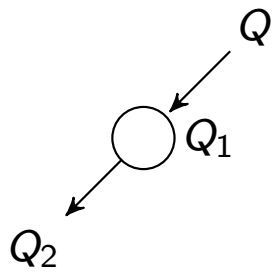
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query
    Q=animal id who ate 1 day before;
    Q_2=send eating animal's id and corresponding time;
    Q_1=store animal's last eating time and filter for sending;
    Lower node sends all data continuously, upper node stores and filters

Proposed Architecture

Query-Based System

- Each node runs a query on some data



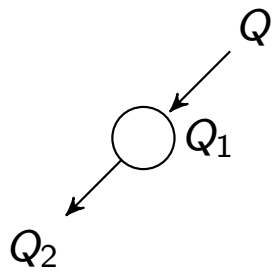
$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query
    Q=animal id who ate 1 day before;
    Q_2=send eating animal's id and corresponding time;
    Q_1=store animal's last eating time and filter for sending;
    Lower node sends all data continuously, upper node stores and filters
    (Historical query)

Proposed Architecture

Query-Based System

- Each node runs a query on some data



$$Q = Q_1 \circ Q_2$$

- Query splitting based on capabilities.
  For example,
  - Sick animal Query
    Q=animal whose body_temp>102;
    Q_2=animal whose body_temp>102;
    Q_1=identity;
    Lower node filters and sends continuously, upper node resends
    (Instantaneous query)
  - Starving animal Query
    Q=animal id who ate 1 day before;
    Q_2=send eating animal's id and corresponding time;
    Q_1=store animal's last eating time and filter for sending;
    Lower node sends all data continuously, upper node stores and filters
    (Historical query)
- Novel concept of query composition with data aggregation

Emergent Systems

Emergent Systems

A kind of intelligence needed where:

Emergent Systems

A kind of intelligence needed where:

- Individual nodes perform simple tasks

Emergent Systems

A kind of intelligence needed where:

- Individual nodes perform simple tasks
- Multiple nodes interact to perform complex tasks

Emergent Systems

A kind of intelligence needed where:

- Individual nodes perform simple tasks
- Multiple nodes interact to perform complex tasks

Emergence

Emergent Systems

A kind of intelligence needed where:

- Individual nodes perform simple tasks
- Multiple nodes interact to perform complex tasks

Emergence: Complex systems arising out of a lot of simple interactions

Transport Layer

Challenges

Transport Layer

Challenges

- Impact of Re-estimation of Route:

Transport Layer

Challenges

- Impact of Re-estimation of Route:
  - Routes can break due to mobility, node failure, etc.

Transport Layer

Challenges

- Impact of Re-estimation of Route:
  - Routes can break due to mobility, node failure, etc.
  - Throughput reduces after every route breakage
- Frequent Congestion

Transport Layer

Challenges

- Impact of Re-estimation of Route:
  - Routes can break due to mobility, node failure, etc.
  - Throughput reduces after every route breakage
- Frequent Congestion
- High Latency

Transport Layer

Challenges

- Impact of Re-estimation of Route:
  - Routes can break due to mobility, node failure, etc.
  - Throughput reduces after every route breakage
- Frequent Congestion
- High Latency
- Energy efficient reliability and full utilization of resources

Transport Layer

Approaches and Improvements

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

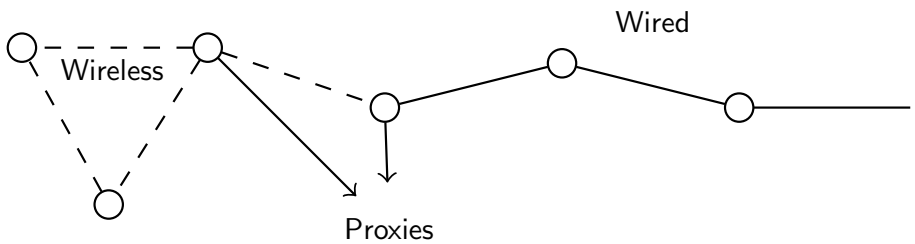
- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks
- Split TCP with improvements:

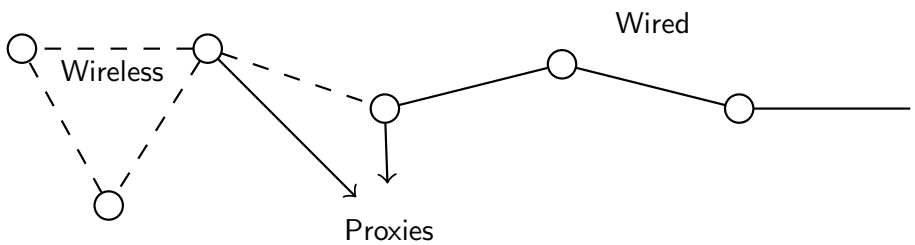


Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks
- Split TCP with improvements:
  - Performance gain through gateways acting as proxy

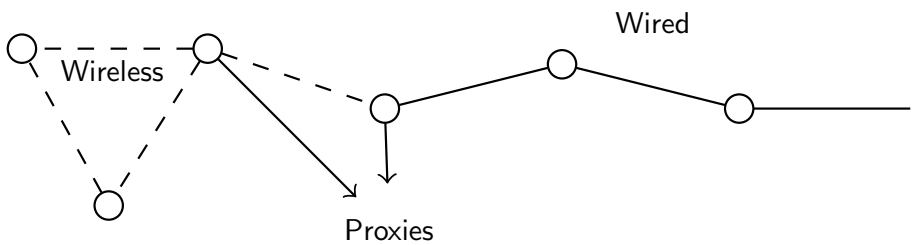


Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks
- Split TCP with improvements:
  - Performance gain through gateways acting as proxy
  - Congestion Control handled by using Buffer at the proxy

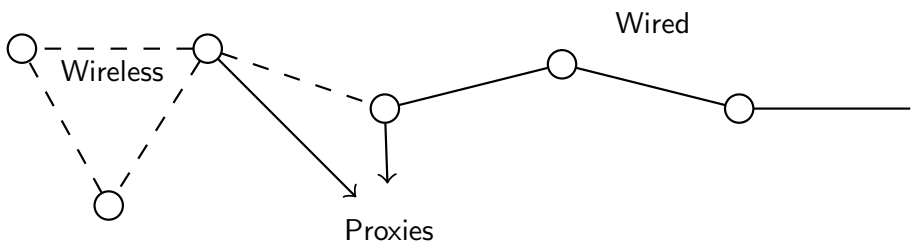


Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks
- Split TCP with improvements:
  - Performance gain through gateways acting as proxy
  - Congestion Control handled by using Buffer at the proxy



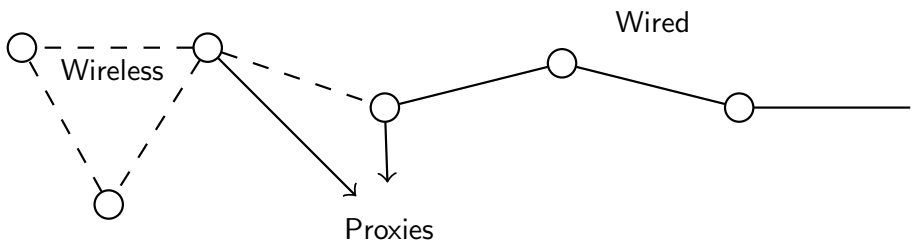
- End-to-end approach:

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks
- Split TCP with improvements:
  - Performance gain through gateways acting as proxy
  - Congestion Control handled by using Buffer at the proxy



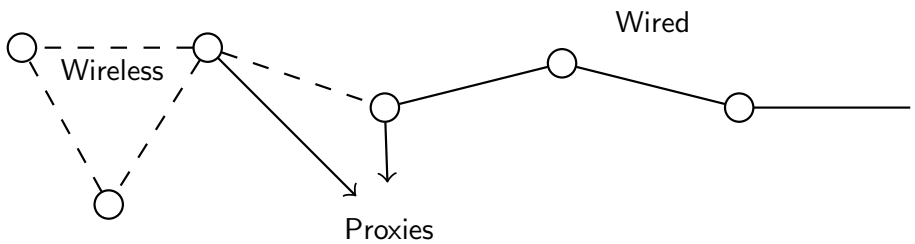
- End-to-end approach:
  - Deal with losses using Selective ACKs

Transport Layer

Approaches and Improvements

Reliability vital for commercial/enterprise applications

- Dual Mode operation with Transport Layer enabled or disabled
- TCP: Though time-tested, a very heavy weight protocol
- Protocol needed for wired-cum-wireless networks
- Split TCP with improvements:
  - Performance gain through gateways acting as proxy
  - Congestion Control handled by using Buffer at the proxy



- End-to-end approach:
  - Deal with losses using Selective ACKs
  - Cumulative ACK(sequence number of nth contagious block)